UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/069,805	11/07/2002	Gilbert Wolrich	10559-307US1	7118

7590 02/07/2007

Fish & Richardson
225 Franklin Street
Boston, MA 02110-2804

EXAMINER	
HUISMAN, DAVID J	
ART UNIT	PAPER NUMBER
2183	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	02/07/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/069,805	WOLRICH ET AL.	
	Examiner	Art Unit	
	David J. Huisman	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *14 November 2006*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-15* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-15* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *14 September 2006* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-15 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: RCE and Amendment as received on 11/14/2006.

Continued Examination Under 37 CFR 1.114

3. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on November 14, 2006, has been entered.

Claim Objections

4. Claim 1 objected to because of the following informalities: In line 3, please insert a comma after "executed". Appropriate correction is required.

5. Claim 2 is objected to because it uses acronyms without first defining what the acronym stands for.

Claim Rejections - 35 USC § 101

6. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

7. Claims 1-9 and 13-15 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

8. Regarding claim 1, the claim is non-statutory for two reasons:

a) the claim calls for "a context branch instruction that, when executed, causes a data processing apparatus to cause an instruction stream to branch...". This language does not include a positive recitation that the branch actually happens. For instance, with a conditional branch, a preceding evaluation of a condition, by itself, ultimately causes the branch instruction to actually branch. However, the evaluation itself is not a tangible result. The examiner recommends replacing "cause an instruction stream to branch to another instruction of the instruction stream" with --branch to another instruction of an instruction stream--. This way, the data processing apparatus will be understood to actually perform a branch as opposed to just performing a condition evaluation which causes a branch, which does not produce a tangible result.

b) for the claimed context branch instruction, branching will occur in accordance with an evaluation, but when the branch does not occur, all that occurs is the evaluation, which is not a tangible result. For instance, take the case of applicant's context branch instruction being the "br=ctx[ctx#..." instruction. For this instruction, an evaluation will be made as to whether ctx# equals a current context number. If the numbers match, then a branch will occur. But, if a match does not occur, then nothing happens in addition to the evaluation. Consequently, since evaluating does not produce a tangible result, the claim as a whole does not produce a tangible result. It follows that claim 1 is non-statutory for this reason.

9. Regarding claims 2, applicant claims a textual instruction format, which is an abstract idea that does not result in functionality being realized by the processor. Claim 2 also does not correct the deficiency of claim 1 because claim 2 is directed towards an instruction format and does not produce a tangible result. Therefore, claim 2 is directed towards non-statutory subject matter.

10. Regarding claim 3, the claim merely expands on the format of the branch instruction, but still does not produce a tangible result. Therefore, claim 3 is directed towards non-statutory subject matter.

11. Regarding claim 4, the claim defines possible values for a context number, but still does not produce a tangible result. Therefore, claim 4 is directed towards non-statutory subject matter.

12. Regarding claim 5, the claim merely expands on the format of the branch instruction, but still does not produce a tangible result. Therefore, claim 5 is directed towards non-statutory subject matter.

13. Regarding claim 6, applicant claims "an optional token that causes a processor to execute..." For the same reasons set forth in the rejection of claim 1 above, applicant has not claimed that the execution takes place, but instead claims a token for causing execution. Therefore, no tangible result is produced, and consequently, claim 6 is directed towards non-statutory subject matter.

14. Regarding claim 7, for the claimed context branch instruction, branching will occur in accordance with an evaluation, but when the branch does not occur, all that occurs is the evaluation, which is not a tangible result. For instance, take the case of applicant's context

branch instruction being the "br=ctx[ctx#..." instruction. For this instruction, an evaluation will be made as to whether ctx# equals the context number of an executing context. If the numbers match, then a branch will occur. But, if a match does not occur, then nothing happens in addition to the evaluation. Consequently, since evaluating does not produce a tangible result, the claim as a whole does not produce a tangible result. It follows that claim 7 is non-statutory for this reason.

15. Regarding claim 8, this claim is non-statutory for the same reasons set forth in the rejection of claim 7 above. That is, since claim 8 claims "branching **if** the executing context number matches the specified context number," it is non-statutory because there is no tangible result produced when the executing context number does not match the specified context number.

16. Regarding claim 9, the claim defines possible values for a context number, this definition does not produce a tangible result. Consequently, claim 9 is directed towards non-statutory subject matter.

17. Regarding claim 13, for the claimed branch instruction, branching will occur in accordance with an evaluation, but when the evaluation results in the branch not occurring, all that occurs is the evaluation, which is not a tangible result. For instance, take the case of applicant's branch instruction being the "br=ctx[ctx#..." instruction. For this instruction, an evaluation will be made as to whether ctx# equals the context number of an executing context. If the numbers match, then a branch will occur. But, if a match does not occur, then nothing happens in addition to the evaluation. Consequently, since evaluating does not produce a

tangible result, the claim as a whole does not produce a tangible result. It follows that claim 13 is non-statutory for this reason.

18. Regarding claim 14, this claim is non-statutory for the same reasons set forth in the rejection of claim 13 above. That is, since claim 14 claims that "a branch occurs **if** the executing context number matches the specified context number," it is non-statutory because there is no tangible result produced when the executing context number does not match the specified context number.

19. Regarding claim 15, the claim defines possible values for a context number, this definition does not produce a tangible result. Consequently, claim 15 is directed towards non-statutory subject matter.

Claim Rejections - 35 USC § 112

20. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

21. Claim 6 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

22. Claim 6 recites the limitation "the branch operation" in line 4. There is insufficient antecedent basis for this limitation in the claim. For purposes of examination, the examiner will interpret "the branch operation" as --a branch operation--.

Withdrawn Rejections

23. Applicant, by way of amendment and arguments, has overcome the prior art rejections set forth in the previous Office Action for claims 1-15. Consequently, these rejections are hereby withdrawn by the examiner, and applicant's arguments filed on November 14, 2006, while considered, are moot. However, upon further consideration, a new ground(s) of rejection is applied below.

Claim Rejections - 35 USC § 102

24. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

25. Claims 1, 7-9, and 13-15 are rejected under 35 U.S.C. 102(e) as being anticipated by Saulsbury et al., U.S. Patent No. 6,314,510 (herein referred to as Saulsbury).

26. Referring to claim 1, Saulsbury has taught a computer program product residing on a computer readable storage medium (Fig.2, component 102; column 2, lines 42-45) comprising instructions (Fig.4), including a context branch instruction (Fig.4 and column 4, lines 44-45; note the "branch on zero" (bz) instruction) that, when executed, causes a data processing apparatus to cause an instruction stream to branch to another instruction of the instruction stream associated with a label specified by the context branch instruction (Fig.4; note that the "bz next1" instruction will branch to the instruction associated with label "next1", which is the "btst wr1,

db1" instruction) based on an evaluation of whether a current context number matches a context number specified by the context branch instruction. The bz instruction is a branch on zero instruction, which means that if the context number specified by the branch instruction (zero) matches a current context number (i.e., the value to be compared to zero), then a branch will occur. In this program, the "bz next1" instruction checks to see if the current context number (dirty bit db0) is equal to 0, the specified context number. See column 4, lines 39-45. It should be noted that the branch is a context branch and the numbers are context numbers as the branches and numbers are associated with contexts (i.e., executing environments). See the title, abstract, and column 2, lines 40-41.

27. Referring to claim 7, Saulsbury has taught a method of operating a processor comprising:

a) evaluating a context number of an executing context to determine whether the context number of the executing context matches a context number specified by a context branch instruction. See the "bz next1" instruction of Fig.4 and column 4, lines 44-45. This instruction evaluates an executing context number (dirty bit db0) to see if it equals the context number specified by the branch instruction, i.e., 0, since it is a "branch if 0" instruction. See column 4, lines 39-45. Note that the branch is a context branch and the numbers are context numbers as the branches and numbers are associated with contexts (i.e., executing environments). See the title, abstract, and column 2, lines 40-41.

b) branching to a specified instruction in accordance with evaluating the context number of the executing context. See Fig.4, and column 4, lines 39-45. Being a "branch if zero" instruction, if the dirty bit db0 equals 0, then the system will branch to the instruction located at the address "next1".

28. Referring to claim 8, Saulsbury has taught a method as described in claim 7. Saulsbury has further taught branching if the executing context number matches the specified context number. As discussed above, and as shown in Fig.4 and column 4, lines 39-45, if the dirty bit db0 is evaluated and determined to be equal to 0, which is the condition specified by the bz instruction, then a branch will occur.

29. Referring to claim 9, Saulsbury has taught a method as described in claim 7. Saulsbury has further taught that the context number has valid values of 0, 1, 2, or 3. Again, see the bz instruction of Fig.4. Since the bz instruction is "branch if 0", 0 is a valid specified context number.

30. Referring to claim 13, Saulsbury has taught a computer program product residing on a computer-readable storage medium (Fig.2, component 102; column 2, lines 42-45), for causing a processor that executes multiple contexts to perform a function, comprises instructions (Fig.4) causing the processor to:

a) evaluate a context number of an executing context to determine whether the context number of the executing context matches a context number specified by a branch instruction. See the "bz next1" instruction of Fig.4 and column 4, lines 44-45. This instruction evaluates an executing context number (dirty bit db0) to see if it equals the context number specified by the branch instruction, i.e., 0, since it is a "branch if 0" instruction. See column 4, lines 39-45. Note that the branch is a context branch and the numbers are context numbers as the branches and numbers are associated with contexts (i.e., executing environments). See the title, abstract, and column 2, lines 40-41.

b) branch to a specified instruction in accordance with evaluating the context number of the executing context. See Fig.4, and column 4, lines 39-45. Being a "branch if zero" instruction, if the dirty bit db0 equals 0, then the system will branch to the instruction located at the address "next1".

31. Referring to claim 14, Saulsbury has taught a product as described in claim 13. Saulsbury has further taught that a branch occurs if the executing context number matches the specified context number. As discussed above, and as shown in Fig.4 and column 4, lines 39-45, if the dirty bit db0 is evaluated and determined to be equal to 0, which is the condition specified by the bz instruction, then a branch will occur.

32. Referring to claim 15, Saulsbury has taught a product as described in claim 13. Saulsbury has further taught that the context number has valid values of 0, 1, 2, or 3. Again, see the bz instruction of Fig.4. Since the bz instruction is "branch if 0", 0 is a valid specified context number.

Claim Rejections - 35 USC § 103

33. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

34. Claims 2-6 are rejected under 35 U.S.C. 103(a) as being unpatentable over Saulsbury in view of Perets et al., U.S. Patent No. 6,564,316 (herein referred to as Perets).

35. Referring to claim 2, Saulsbury has taught a computer program product as described in claim 1. Saulsbury has further taught that the branch instruction is of the format br=ctx[ctx#, label#], which is interpreted as "branch if equal to a context number (br=ctx), where the context number (ctx#) is 0 and the branch destination is next1 (label#)". Again, see Fig.4 (bz instruction). Saulsbury has not taught that the instruction format includes an optional token. However, Perets has taught a branch instruction which includes an optional token. See Fig.6, step 600, and column 5, line 66, to column 6, line 5. The programmer-optional field (delay slot field) is used to specify the amount of delay slot instructions, where the use of delay slots for execution significantly speeds up the execution time of branch instructions. See column 5, lines 59-62. As a result, in order to speed up execution, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Saulsbury to include the option token (delay slot field) of Perets.

36. Referring to claim 3, Saulsbury in view of Perets has taught a computer program product as described in claim 2. Saulsbury has further taught that the label# is a symbolic label corresponding to an address of the other instruction, and wherein ctx# is the context number. Again, "next1" (Fig.4) is the label (address) of the destination instruction, as is known, while ctx# is the context number that the branch is concerned with (in this case, the branch is concerned with 0).

37. Referring to claim 4, Saulsbury in view of Perets has taught a computer program product as described in claim 3. Saulsbury has further taught that the specified context number has valid values of 0, 1, 2, or 3. See the bz instruction of Fig.4. Since the bz instruction is "branch if 0", 0 is a valid specified context number.

38. Referring to claim 5, Saulsbury has taught a computer program product as described in claim 1. Saulsbury has not taught that the branch instruction has an optional token. However, Perets has taught a branch instruction which includes an optional token. See Fig.6, step 600, and column 5, line 66, to column 6, line 5. The programmer-optional field (delay slot field) is used to specify the amount of delay slot instructions, where the use of delay slots for execution significantly speeds up the execution time of branch instructions. See column 5, lines 59-62. As a result, in order to speed up execution, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Saulsbury to include the option token (delay slot field) of Perets.

39. Referring to claim 6, Saulsbury in view of Perets has taught a computer program product as described in claim 5. Perets has further taught that the instruction has an optional token that causes a processor to execute a number of instructions corresponding to the value of the optional token following the branch instruction before performing the branch operation. See column 3, lines 3-7.

40. Claims 10-12 are rejected under 35 U.S.C. 103(a) as being unpatentable over Saulsbury.

41. Referring to claim 10, Saulsbury has taught a processor that can execute multiple contexts and that comprises:

a) a register stack. See Fig.1, component 110.

b) Saulsbury has taught an execution unit (Fig.1, component 108) coupled to the register stack and a program control store that stores a context branch instruction (Fig.1, component 102, and column 2, lines 42-45) that causes the processor to:

b1) evaluate a context number of an executing context to determine whether the context number of the executing context matches a context number specified by the branch instruction. See the "bz next1" instruction of Fig.4 and column 4, lines 44-45. This instruction evaluates an executing context number (dirty bit db0) to see if it equals the context number specified by the branch instruction, i.e., 0, since it is a "branch if 0" instruction. See column 4, lines 39-45. Note that the branch is a context branch and the numbers are context numbers as the branches and numbers are associated with contexts (i.e., executing environments). See the title, abstract, and column 2, lines 40-41.

b2) branch to a specified instruction in accordance with evaluating the context number of the executing context. See Fig.4, and column 4, lines 39-45. Being a "branch if zero" instruction, if the dirty bit db0 equals 0, then the system will branch to the instruction located at the address "next1".

c) Saulsbury has not explicitly taught a program counter for each executing context. However, this limitation is deemed to be inherent by the examiner. A program counter must exist because the program counter is the component that holds the address of the next instruction to be fetched. Without the PC, the system would not be able to fetch an instruction, which means that no work would be done. A program counter has to exists for each context because each context includes instructions that need to be fetched and executed.

d) Saulsbury has not taught that the execution unit is an arithmetic logic unit. However, Official Notice is taken that ALUs and their advantages are well known execution units that are expected in the art. An ALU is the term given to a unit that is able to perform arithmetic operations (addition, subtraction, etc.) and logical operations (AND, OR, NOT, etc.). Clearly, by

implementing an ALU, the system would be able to perform a variety of operations on data. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the execution unit of Saulsbury to be an ALU so that Saulsbury is able to perform both arithmetic and logical operations.

42. Referring to claim 11, Saulsbury has taught a processor as described in claim 10. Saulsbury has further taught that a branch occurs if the executing context number matches the specified context number. As discussed above, and as shown in Fig.4 and column 4, lines 39-45, if the dirty bit db0 is evaluated and determined to be equal to 0, which is the condition specified by the bz instruction, then a branch will occur.

43. Referring to claim 12, Saulsbury has taught a processor as described in claim 10. Saulsbury has further taught that the context number has valid values of 0, 1, 2, or 3. Again, see the bz instruction of Fig.4. Since the bz instruction is "branch if 0", 0 is a valid specified context number.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (571) 272-4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

DJH
David J. Huisman
January 18, 2007